APPLICATION FOR UNITED STATES LETTERS PATENT

Applicants: Dmitry Andreev, Gregory Vilshansky and Boris Vishnevsky

For: SYSTEM AND METHOD OF PROVIDING CREDENTIALS IN A NETWORK

Docket No.: END920030143

INTERNATIONAL BUSINESS MACHINES CORPORATION

CERTIFICATE OF MAILING UNDER 37 CFR 1.10

I hereby certify that, on the date shown below, this correspondence is being deposited with the United States Postal Service in an envelope addressed to the Assistant Commissioner for Patents, Washington, D.C., 20231 as A Express Mail Post Office to Addressee@

Mailing Label No. EJ805276154US

On 3/2/04

June Mitchell

Name of person mailing paper

June Mitchell 3/2/04

Signature Date

SYSTEM AND METHOD OF PROVIDING CREDENTIALS IN A NETWORK

DESCRIPTION

BACKGROUND OF THE INVENTION

Field of the Invention

The invention generally relates to a system and method of providing credentials in a network, in particular, providing short living credentials to a target application without using persistent memory.

Background Description

Providing friendly and simplified access to networks may often be at odds with providing protection of a user's identity and password(s). Security breaches have become a significant problem in networks when fraudulent acquisition of users' identities and passwords occurs. Fraudulent access to network resources has caused a plethora of damage which has forced network and application providers to increase security methods and measures.

Spoofing a user's identity has been a constant threat and often occurs while an authentic user is accessing a network resource. When authentication parameters are in

the process of being analyzed and/or validated, an opportunity may exist for a security breach. Avoiding a security breach and detecting when a security breach has occurred has been an ongoing endeavor. However, security breaches are still prevalent.

For example, in a network system, to provide single sign on (SSO) functionality, a portal server may need to submit a user's ID and password to other applications. Storing the user's confidential information in a persistent storage inherently poses a security threat since the information would be present for potential unauthorized access.

Furthermore, many security standards require that passwords are not stored in non-persistent memory longer than it is absolutely necessary to perform the user authentication, i.e., password image in memory must be destroyed immediately after its has been used. Thus, even in the case when the target application (TA) uses the same information as those used by the portal, and the logon to the TA does not happen immediately after logon to the portal, but rather at the time of the end user's choosing, to provide a secure SSO mechanism, a solution without storing user's information in memory is highly desirable.

Creation of a security process that avoids inordinate exposure of a user's confidential password to downstream network resources would be a desirable improvement to network security. Further, if authentication of a user during significant stages of network access is capable of detecting a fraudulent use of a user's information, then an increase in overall security may be achieved.

SUMMARY OF THE INVENTION

In an aspect of the invention, a method is provided for authentication in a network. The method comprises creating a credential string which is derived from a session ID, and sending a UserID associated with the session ID and the credential string to a software application. The method also includes receiving a confirmation request which includes the credential string from an authentication proxy, and sending a response in reply to the confirmation request to the authentication proxy based upon the validity of the credential string.

In another aspect of the invention a method for authenticating a user request for a software application is provided. The method comprises the steps of receiving a UserID and credential string at an authentication proxy server and sending a confirmation request from the authentication proxy to a portal, the confirmation request includes the credential string. The method further includes the steps of receiving a response at the authentication proxy for the confirmation request and validating the UserID using a light weight directory access protocol (LDAP) lookup request and the response.

In another aspect of the invention, a system for authenticating a session is provided. The system comprises an authentication proxy which receives requests to authenticate a UserID and credential string and a credential string validation component which receives requests to validate the credential string, wherein the

credential string validation component checks whether the credential string has been previously received for validation within a predetermined time period.

In another aspect of the invention, a computer program product is provided comprising a computer usable medium having readable program code embodied in the medium and including at least one program code to create a credential string which is derived from a session ID, send a UserID associated with the session ID and the credential string to a software application, receive a confirmation request which includes the credential string from an authentication proxy and send a response in reply to the confirmation request to the authentication proxy based upon the validity of the credential string.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1A is a block diagram showing components of an embodiment of the invention;

Figure 1B is a block diagram showing components of an embodiment of the invention;

Figure 2 is a flow diagram showing steps of an embodiment of the invention; and

Figures 3Aand 3B are flow diagrams showing steps of an embodiment of the invention.

DETAILED DESCRIPTION OF EMBODIMENTS OF THE INVENTION

This invention is directed to a system and method of providing authentication to applications without using persistent memory to jeopardize the security of the authentication process. In one aspect, this is accomplished by using a short-living credential string, generated by and verifiable by a portal, for authenticating a user to a target application (TA). For example, in a scenario when a system integrator, or the like, has access to the link between the TA and its authentication authority (e.g., a lightweight directory access protocol (LDAP) server), a secure solution based on an intelligent proxy server is provided. By using the invention, the need for exposing a user's password to network resources beyond a portal may be avoided.

Figures 1A and 1B is an exemplary block diagram showing components of an embodiment of the invention. Figures 1A and 1B may also be a flow chart illustrating steps of using the invention. The steps are indicated by reference numerals S200-S240, as explained below.

The embodiment of Figure 1A includes a network 100 (such as, for example, the Internet) connecting one or more client computers 105 (e.g., a personal computer (PC)) with a portal server 110, which typically includes an application containing at least a personalized page of links to target applications. Portals may include pages containing data dynamically collected from target applications. The portal server 110

may also include a credential string validator 112, explained below. The portal server 110 may be in communication with at least one LDAP Proxy 115 for authentication purposes. In general, the proxy server, e.g., 115, which may include a software module that intercepts session communications between a client module (e.g., a module on client 105) and a server module (e.g., TA 130 or LDAP 125), may be capable of either passing requests and responses through, or performing an additional processing and/or modification of requests and/or responses. The proxy server may be transparent to both the client and the server modules.

The LDAP Proxy 115 may be in communication with the LDAP server 125. The TAs 130 may include an IP network product requiring user authentication and session tracking. The TAs 130 may be in communication with the LDAP Proxy 115. The interconnectivity among these components (e.g., 110, 115, 125, and 130) may be accomplished using one or more network(s) 135, which may be a local area network (LAN), a wide-area network (WAN), or the like.

In an aspect of the invention, a session may include a sequence of HTTP requests (or similar protocol requests) and responses traversing between a client browser on a client computer (e.g., 105) and a target application (e.g. 110) or a portal server (e.g.130), beginning with sign-on and ending with either closing of a browser window or an explicit logoff operation. A logon by a user may include a client authentication, involving entering user-specific credentials, examples of which are a user ID and a password, for the purpose of gaining access to a target application (e.g., 130). A single sign-on (SSO) typically may include functionality

that allows a user to access target applications (e.g., 130) linked to the portal (e.g., 110) without re-entering target application specific information such as, for example, user ID and password during a session initiated by successful authentication of the user with the portal. In use, the password is not propagated through the system to verify and authenticate the user's access. Instead, a credential string is used in order to provide additional security to the system. In one embodiment, the credential string may be a hash of the session ID, which may be encrypted.

By way of example, and referring to Figure 1A, at step S200, a user successfully logs onto the portal 110 from a client computer 105, and wishing to initiate a session with a TA 130 (usually by clicking the appropriate URL link within the client browser). At step S210a, the portal 110 sends a session ID derivative (e.g., a hash of the session ID) as a credential string, instead of the user's password to the TA 130. At step S220, the TA 130 tries to authenticate the user, e.g., by sending a BIND request to a LDAP directory specifying the received credential string as the LDAP required parameter. The LDAP Proxy 115 intercepts the BIND request and validates (i.e., authenticates) the user's credentials.

At step S225, the UserID is validated via a LDAP 125 lookup request. At step S230, the password is validated via the credential string validator component 112, which upon receiving the password validation request from the LDAP proxy 115, confirms with the portal 110 that the credential string presented has been actually produced from the active session ID, and the session belongs to the same user. At step

S240, the LDAP proxy 115 returns an authentication confirmation (e.g., "BIND successful") to the TA 130 if both the credential string and UserID are independently confirmed. The arrows of Figure 1A and 1B associated with steps S200-S240 may also be considered bi-directional communication paths.

For general security considerations, a new credential string must be generated by the portal 110 for every user's request to access a TA. Since, in some portal implementations, the session ID may be reconstructed from HTTP traffic between the portal and the client browser, it may be necessary to use encryption when producing the credential string based on session ID. In some network configurations and situations, it may be possible for an intruder to intercept the short-living credential string in transit between the portal 110 and the TA 130. Therefore, the communications between the portal server 110 and the TA 130 may optionally use an encrypted transport layer. A password, however, is no longer required to propagate throughout the network, in accordance with the invention.

Figure 1B is a block diagram showing components of an embodiment of the invention. This embodiment includes at least one TA Proxy 120 in addition to the components of Figure 1A. The TA Proxy 120 may be a software component resident in the one or more TAs 130 or in any other suitable server. The at least one TA Proxy 120 is in communication with the one or more TA's 130 and the portal server 110

By way of example, and referring to Figure 1B, to further improve security, at step S200, a user successfully logs on to the portal 110 from a client computer 105 and wishes to initiate a session with a TA 130 (usually by clicking the appropriate URL

link within the client browser). At step S210b, the portal 110 sends a session ID derivative as a credential string and UserID, produced as an encrypted hash of the active user's session ID, to the TA Proxy 120 where the TA Proxy checks whether the UserID and credentials have been received recently (i.e., a predetermined time period). If not received recently, then at step S215, the UserID and credential string are sent to the TA 130. If, however, the UserID and credential string have been recently received and, at step 235, a second request with the same UserID and credential string is received by TA Proxy 120, procedures associated with a network security breach may be initiated. The procedures may include terminating all the sessions authenticated by the compromised credential string.

At step S220, the TA 130 attempts to authenticate the user, e.g., by sending a BIND request to a LDAP directory specifying the received credential string as the LDAP password. The LDAP Proxy 115 intercepts the BIND request and validates (i.e., authenticates) the user's credentials. At step S225, the UserID may be validated via a LDAP 125 lookup request. At step S230, the password is validated via the credential string validator component 112, which upon receiving the password validation request from the LDAP proxy 115, confirms with the portal 110 that the credential string, presented instead of the user's password, has been actually produced from the active session ID, and the session belongs to the same user. The credential string validator module 112 validates each short-living credential string only once, and upon detecting a second validation request for the same credential string, sends a message to the TA proxy 120 informing it about the attempt to breach security (e.g.,

short-living credential being intercepted). The credential string validator module 112 may also initiate any procedures typically associated with a network security breach. Typically, the TA proxy 120 immediately terminates all the sessions authenticated by the compromised credential string in such instances.

Two security breach situations may be typical: (a) more probable, an authentication request to the TA, initiated by an intruder using a spoofed credential string, may be served after an "already-in-progress" authentication request from the portal 110, and (b) a situation may be possible where the authentication request to the TA 130, initiated by the intruder using a spoofed credential string, may be served before the "already-in-progress" authentication request from the portal 110. In situation (a), the intruder's session is not initiated. In situation (b), the intruder may be able to initiate a user session with the TA 130. However, in the scenario of (b), the session still terminates when the "already-in-progress" authentication request from the portal 110 is presented to the credential string validator module 112 by the second requestor.

Using the Invention

Figure 2 is a flow diagram showing steps of an embodiment of the invention, starting at step 300. Figure 2 as well as Figures 3A and 3B, may equally represent a high-level block diagram of components of the invention implementing the steps thereof. The steps of Figures 2-3B may be implemented on computer program code in combination with the appropriate hardware. This computer program code may be

stored on storage media such as a diskette, hard disk, CD-ROM, DVD-ROM or tape, as well as a memory storage device or collection of memory storage devices such as read-only memory (ROM) or random access memory (RAM).

Continuing with the flow of Figure 2, at step 305, a user establishes a session with a portal server. At step 310, a request is made to obtain the UserID and password from the user. At step 315, an authentication check is made by the portal to determine whether the UserID and password is valid. If not valid, another request is made at step 310. Otherwise, if successful, at step 320, the user chooses an application which may be from a list presented in the user's browser.

At step 325, the portal creates a credential string as a hash of user's SessionID. At step 330, the portal server sends the UserID and credential string to the target application selected by the user. At step 335, the target application attempts to authenticate (e.g., a BIND request) against the LDAP Proxy server by sending the UserID and the credential string to the LDAP Proxy. At step 340, the LDAP proxy performs LDAP lookup against the LDAP server for the UserID.

At step 345, a check is made whether the lookup was successful. If not successful, at step 365 the session is terminated. Otherwise, if successful, at step 350, the LDAP proxy authenticates the credential string against a credential string validation module. At step 355, a check is made whether the check is successful and, if not, at step 365, the authentication fails and the session is terminated. Otherwise if successful, at step 360, the session is established (e.g., a "BIND Successful" returned to the target application). The process completes at step 370.

Referring to Figures 3A and 3B, a process starts at step 400. At step 405, a user establishes a session with a portal server. At step 410, a request to obtain the UserID and password from the user is presented. At step 415, a check is made to authenticate the password and UserID. If the authentication fails, then the request is re-issued at step 410. Otherwise, if successful, at step 420, the user selects a target application which may be from a list presented in the browser.

At step 425, the portal creates a credential string as a hash of the user's sessionID. At step 430, the portal server sends the UserID and credential string to the target application proxy. At step 435, the target application proxy intercepts the UserID and credential string and checks whether the UserID and credential string has been recently used. The time period to determine whether it has been recent is an alterable parametric that may have a wide range of values suitable for a wide-range of network operations, for example, from a few milliseconds to several minutes or hours/days. If recently used, then at step 455, a security breach is initiated. If not recently used, then at step 440, the target application proxy sends UserID and credential string to the selected target application.

At step 445, the target application attempts to authenticate the user against the LDAP server by sending the UserID and credential string to the LDAP proxy. At step 450, the LDAP proxy performs a LDAP lookup against the LDAP server to validate the UserID. At step 460, if the lookup is successful, then at step 465, the LDAP proxy authenticates the credential string by sending it to the credentials validation module typically associated with the portal server. If the authentication succeeds, then at step

475, a "BIND successful" response is sent to the target application and the session is established between user and target application.

If at step 460, the lookup is unsuccessful, or if, at step 470, the authentication is unsuccessful, or a security breach at step 480, a bad credentials message is returned and the session(s) with the UserID is/are terminated.

While the invention has been described in terms of embodiments, those skilled in the art will recognize that the invention can be practiced with modifications and in the spirit and scope of the appended claims.